Articulating The Need For Integrated Model-Centric Engineering

Todd Bayer

Principal Engineer
Jet Propulsion Laboratory
California Institute of Technology

What Problems are We Trying to Fix?

- There have been several expositions recently regarding problems in systems and software engineering:
  - [excised]

- In 2009 IMCE studied these in order to refine its understanding of the problems and establish some concrete goals for what to fix.
  - Five themes emerged from this study (IOM 3100-09-040)
  - Described in the following material

Five System Engineering Problem Areas

1. Mission complexity is growing faster than our ability to manage it
   …increasing mission risk from inadequate specification & incomplete verification

2. System design emerges from the pieces, not from an architecture
   …resulting in systems which are brittle, difficult to test, and complex and expensive to operate.

3. Knowledge and investment are lost at project lifecycle phase boundaries
   …increasing development cost and risk of late discovery of design problems.

4. Knowledge and investment are lost between projects
   …increasing cost and risk; damping the potential for true product lines

5. Technical and programmatic sides of projects are poorly coupled
   …hampering effective project decision-making; increasing development risk.

Theme 1:
Growing Risk from Unmanaged Complexity

- As our mission set evolves from flyby reconnaissance to orbital study to in-situ exploration, our missions grow in complexity.
- This is enabled by ever-increasing spaceborne processing power…
- …and by the software running on it
  - More and more of flight system functionality (i.e., behavior) is implemented in software
- But traditional methods of writing requirements are inadequate to capture and expose these system level characteristics
  - Unintended system behaviors often emerge.
- Also, the potential range of behaviors has become so large that it is impractical to fully test it.
  - Unintended system behaviors can no longer be reliably exposed by testing.

- RESULT: inadequately-specified and incompletely-verified system level interactions are a major and growing risk factor for our missions.

In-Flight Problems Often Result

- Mission Ops teams must work around unresolved development issues
  - [excised] is struggling with operability deficiencies due to
    - insufficient understanding of science-engineering trade space
    - lack of agreed orbital scenario during design
  - [excised] had operability problems due to insufficient understanding of EEIS dynamics
- Significant anomalies can occur; some are mission-threatening
  - [excised] attitude control interactions with battery charging resulted in safe mode
  - [excised] experienced several anomalies attributable to unmanaged complexity …

Theme 2:
System Design Emerges from the Pieces

System design emerges from the pieces, rather than from an architected solution, resulting in systems which are brittle, difficult to test, and complex and expensive to operate.

- Architectural principles are seldom articulated or used in design
- Where they exist, principles are abandoned to solve pressing technical problems
- System designs are spread across multiple disconnected artifacts
- Control parameters for a function are scattered across the system
  - [excised] MCS Stellar, UHF Relay anomalies
- Domain physics-based models are not connected to each other or to a system model
- Insufficient consideration of V&V during requirements development
  - [excised] FPGA test software
- Actual science merit of a given point solution is not known until late
  - [excised] operability issues
- Desired system behaviors are poorly articulated, resulting in software whose behavior must be ‘discovered’
  - [excised] in-flight SIB swaps

Theme 3: Investments Lost at Phase Boundaries

Knowledge and investment are lost at project lifecycle phase boundaries, resulting in increased development cost and risk of delayed discovery of design problems.

- Formulation models are abandoned and new ones created when implementation phase begins:
  - System Trades Model, TeamX models, Early Cost models
  - Power Models, Telecom Link Models, Data Transport Models
  - Many other unique ad hoc models
- CM of existing models is lacking, impeding continued use
- Essential attributes of design are not captured consistently in readily accessible manner:
  - Architectural principles
  - Trade study assumptions and rationale
  - System Design
- Training takes longer than necessary
  - Affects staffing arc during phases A-D
  - Affects team turnover as projects moves into operations

Theme 4:
Insufficient Re-use of System Designs

System design re-use is lacking, increasing cost and risk, and damping the potential for true product lines.

- Because system architectures and designs are not well-captured, re-using them on subsequent projects is difficult and seldom happens
  - except where the project team itself is 'inherited' by the next project
- Heritage reviews narrowly focus on full re-use of components
- Too much of the system development "way of doing business" is custom
  - tools (some)
  - models (more)
  - processes(much more)
- The current institutional guidance (e.g., JPL Design Principles), while providing important and useful heuristics and lessons learned, is not sufficient to enable architecture re-use.

Theme 5:
Poor Technical - Programmatic Coupling

Technical and programmatic sides of projects are not well-coupled, hampering effective project decision-making and increasing development risk.

- Cost, schedule, scope, investment, risk implications of a given set of requirements, science objectives, components, functions is very difficult to determine.
  - [excised] scope to budget reconciliation problems
- Trade studies seldom fully incorporate programmatic considerations. Existing tools do not support such a view.

How will Integrated Model-Centric Engineering help solve the five problems?

Current State, Future State

Current

1. Mission complexity is growing faster than our ability to manage it, increasing mission risk from inadequate specification & incomplete verification

Future

- Enhanced understanding of system behavior and reasoning about engineering completeness
- Improved communication and reduced confusion using ‘single source of truth’ information
- Automatically generated human-interpretable documentation provides frequent and authoritative snapshots of system properties
- Design reviews consist largely of model inspection and validation
- System test activities focus on model validation and correlation

Current State, Future State (cont'd)

Current

2. System design emerges from the pieces, not from an architecture, resulting in systems which are brittle, difficult to test, and complex and expensive to operate.

Future

- Having a better way to talk about our systems at an architectural level enables us to do a better architecting job
  - Better separation of concerns, less unnecessary coupling, more coherence of function
- Architectural principles are explicit and enforceable in the design
- Behavior of the system is specified rather than discovered:
  - behavioral models will be created by systems and software teams working together
  - models are executable and formally analyzable to discover logic flaws very early in design process
  - The actual flight software directly and faithfully implements the behavioral models
- Design discussions between subsystems and with systems use common, authoritative representations
- Integrated models enable early validation
  - requirements completeness, operability, performance
- Integration with physics-based models enables more complete design space exploration
- Proposed design changes are expressed, analyzed, and considered by change boards in the system model directly
- Kludges are less necessary and their impact more fully understood
- Missions arrive at the launch pad with more of their architecture intact, reducing operations cost and risk

Current State, Future State (cont'd)

Current

3. Knowledge and investment are lost at project lifecycle phase boundaries, increasing development cost and risk of late discovery of design problems.

Future

- System models evolve and mature from formulation through operations
  - Rapid Mission Architecting and TeamX will eventually draw from the same line-developed model libraries as the implementation team
- Rich capture of design information is enabled: structure, behavior, requirements, and parametrics connected in a unified model
  - In turn, enables more effective training of new team members
- Configuration management of the system design is rigorous, for the first time
- Model repositories enables long term data retention and model re-use
  - Doing organizations manage libraries of CM'd, reusable, domain models

Current State, Future State (cont'd)

Current

4. Knowledge and investment are lost between projects, increasing cost and risk; damping the potential for true product lines

Future

- Architecture and detailed designs are captured in a formalized and repeatable system model
- Once the architecture is captured, it is possible to consider reusing all or part of it
- Well-architected systems have less tightly coupled parts, enabling more reuse

Current State, Future State (cont'd)

Current

5. Technical and programmatic sides of projects are poorly coupled, hampering effective project decision-making; increasing development risk.

Future

- Behavioral, physical, cost and risk models are integrated allowing for an integrated fully-informed approach to system optimization
  - Risk and resource implications of a proposed engineering change will be better understood
  - Engineering impacts of a proposed resource change will be better understood